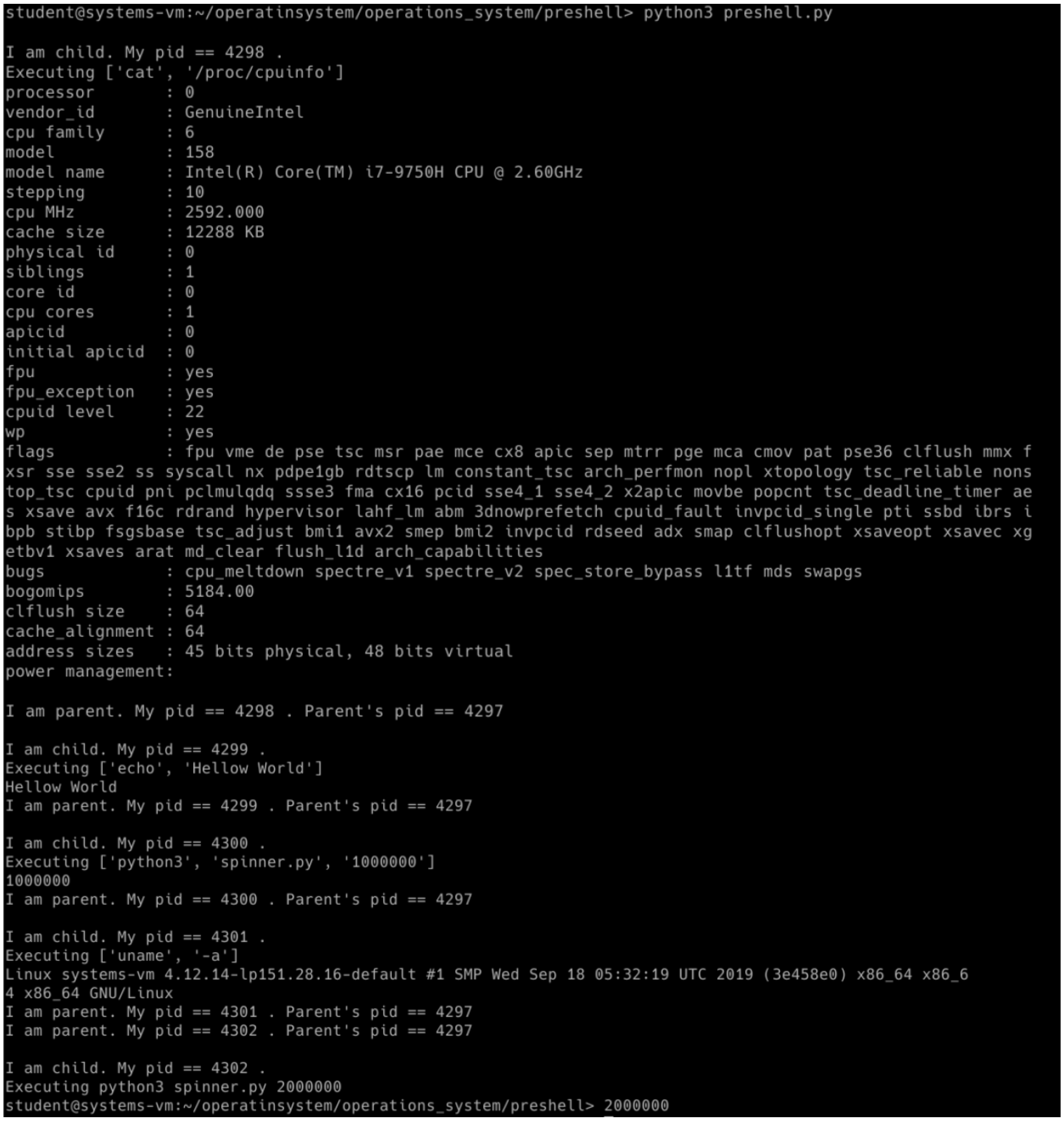
Preshell report Ken Amamori
2/2/2021



```
student@systems-vm:~/operatinsystem/operations_system/preshell> python3 preshell.py

I am child. My pid == 4298 .
Executing ['cat', '/proc/cpuinfo']
processor       : 0
vendor_id       : GenuineIntel
cpu family      : 6
model           : 158
model name      : Intel(R) Core(TM) i7-9750H CPU @ 2.60GHz
stepping        : 10
cpu MHz         : 2592.000
cache size      : 12288 KB
physical id     : 0
siblings        : 1
core id         : 0
cpu cores       : 1
apicid          : 0
initial apicid  : 0
fpu             : yes
fpu_exception   : yes
cpuid level     : 22
wp              : yes
flags           : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush mmx f
xsr sse sse2 ss syscall nx pdpe1gb rdtscp lm constant_tsc arch_perfmon nopl xtopology tsc_reliable nons
top_tsc cpuid pni pclmulqdq ssse3 fma cx16 pcid sse4_1 sse4_2 x2apic movbe popcnt tsc_deadline_timer ae
s xsave avx f16c rdrand hypervisor lahf_lm abm 3dnowprefetch cpuid_fault invpcid_single pti ssbd ibrs i
bpb stibp fsgsbase tsc_adjust bmi1 avx2 smep bmi2 invpcid rdseed adx smap clflushopt xsaveopt xsavec xg
etbv1 xsaves arat md_clear flush_l1d arch_capabilities
bugs            : cpu_meltdown spectre_v1 spectre_v2 spec_store_bypass l1tf mds swapgs
bogomips        : 5184.00
clflush size    : 64
cache_alignment : 64
address sizes   : 45 bits physical, 48 bits virtual
power management:

I am parent. My pid == 4298 . Parent's pid == 4297

I am child. My pid == 4299 .
Executing ['echo', 'Hellow World']
Hellow World
I am parent. My pid == 4299 . Parent's pid == 4297

I am child. My pid == 4300 .
Executing ['python3', 'spinner.py', '1000000']
1000000
I am parent. My pid == 4300 . Parent's pid == 4297

I am child. My pid == 4301 .
Executing ['uname', '-a']
Linux systems-vm 4.12.14-lp151.28.16-default #1 SMP Wed Sep 18 05:32:19 UTC 2019 (3e458e0) x86_64 x86_6
4 x86_64 GNU/Linux
I am parent. My pid == 4301 . Parent's pid == 4297
I am parent. My pid == 4302 . Parent's pid == 4297

I am child. My pid == 4302 .
Executing python3 spinner.py 2000000
student@systems-vm:~/operatinsystem/operations_system/preshell> 2000000
```

```
import os
import sys

def main():
	cmds = [['cat', '/proc/cpuinfo'], ['echo', 'Hellow World'], ['python3', 'spinner.py',
'1000000'], ['uname', '-a']]
	for cmd in cmds:
		rc = os.fork()
		if rc<0:
```

```
                print("Fork failed")
                sys.exit(1)
            elif rc == 0:
                print("\nI am child. My pid ==", os.getpid(), ".\nExecuting", cmd)
                os.execve('/usr/bin/'+cmd[0], cmd, os.environ)
            else:
                rc_wait=os.wait()
                print("I am parent. My pid ==", rc, ". Parent's pid ==", os.getpid())
        last_rc = os.fork()
        if last_rc > 0:
            print("I am parent. My pid ==", last_rc, ". Parent's pid ==", os.getpid())
            exit()
        else:
            print("\nI am child. My pid ==", os.getpid(), ".\nExecuting python3 spinner.py 
2000000")
            os.execv('/usr/bin/python3', ['python3', 'spinner.py', '2000000'])

if __name__ == '__main__':
        main()
```

cmds contains all the commands that need to be executed by the program and within the forloop, it will excute one by one. Since it requires one process(child process) for one command, fork is inside of forloop. If rc is less than 0, failed, if same as 0, successful child process, and greater than 0, the parent process is being running. If rc is 0, it will use execve in order to run the command. Meanwhile, the parent is in wait() since the child will run first. This steps will occurs 4 times for each commands. Afterall, as suggested in the assignment, child process will run python3 spinner.py 2000000', however, the parent will not wait and the program will exit since there are no wait(). Interestingly, the child process yet processes and prints out after it is exited from the program since not deterministic when not using wait().